(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **10055135 A**

(43) Date of publication of application: **24.02.98**

(51) Int. Cl **G09C 1/00**
G09C 1/10

(21) Application number: **08211227**

(22) Date of filing: **09.08.96**

(71) Applicant: **FUJITSU LTD**

(72) Inventor: **KITAJIMA HIRONOBU**
FUEKI SHUNSUKE

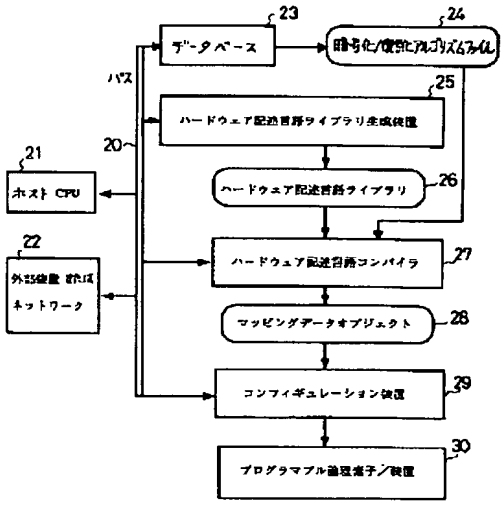
(54) CIPHERING/DECIPHERING DEVICE AND METHOD USING PROGRAMMABLE LOGIC ELEMENT/DEVICE

(57) Abstract:

PROBLEM TO BE SOLVED: To realize a ciphering/deciphering technology which is capable of changing an algorithm flexibly in accordance with the condition of a necessary secrecy or the like and which is of high speed.

SOLUTION: When this device receives changing data, a hardware description language compiler 27 takes out a corresponding ciphering/deciphering algorithm file 24 from a database 23 to compile it by using a hardware description language library 26 generated by a hardware description language library generator 25. A configuration device 29 changes a ciphering/deciphering circuit by writing a mapping data object 28 generated in this manner to a programmable logic element/device 30. Since the constitution of the ciphering/deciphering circuit is automatically changed based on the changing data. the changing of a ciphering/deciphering algorithm is facilitated.

COPYRIGHT: (C)1998,JPO



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－55135

(43)公開日　平成10年(1998)２月24日

(51)Int.Cl.⁶		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０９Ｃ	1/00	６５０	7259－5Ｊ	Ｇ０９Ｃ	1/00	６５０Ｚ
	1/10		7259－5Ｊ		1/10	

審査請求　未請求　請求項の数23　ＯＬ　（全 17 頁）

(21)出願番号　特願平8－211227

(22)出願日　平成８年(1996)８月９日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　北島　弘伸
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(72)発明者　笛木　俊介
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(74)代理人　弁理士　大菅　義之　（外１名）

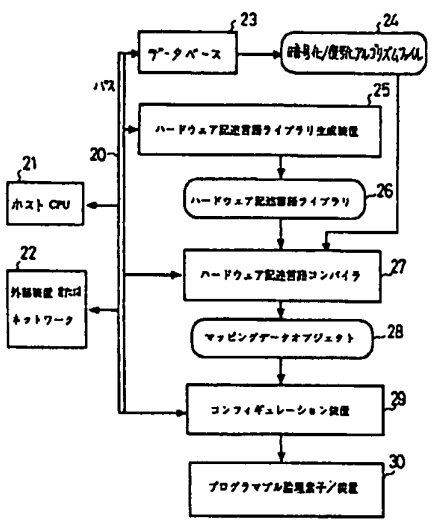
(54)【発明の名称】　プログラマブルな論理素子／装置を用いた暗号化／復号化装置および方法

(57)【要約】

【課題】　必要な機密度などの条件に応じて、フレキシブルにアルゴリズムを変更することが可能で、かつ、高速な暗号化／復号化技術を実現することが課題である。

【解決手段】　変更データを受け取ると、ハードウェア記述言語コンパイラ２７は、対応する暗号化／復号化アルゴリズムファイル２４をデータベース２３から取り出し、ハードウェア記述言語ライブラリ生成装置２５が生成したハードウェア記述言語ライブラリ２６を用いて、それをコンパイルする。コンフィギュレーション装置２９は、こうして生成されたマッピングデータオブジェクト２８をプログラマブル論理素子／装置３０に書き込んで、暗号化／復号化回路を変更する。変更データをもとにして、暗号化／復号化回路の構成が自動的に変更されるので、暗号化／復号化アルゴリズムの変更が容易になる。

暗号化／復号化装置の概略図



【特許請求の範囲】

【請求項１】　少なくとも１つ以上のプログラマブル論理素子を含み、該プログラマブル論理素子を用いて、与えられた暗号化の仕様に対応する暗号化回路を生成する回路手段と、

前記暗号化の仕様を変更するための変更データを読み込み、該変更データに基づいて、前記暗号化回路を自動的に変更する変更手段とを備えることを特徴とする暗号化装置。

【請求項２】　前記変更手段は、前記暗号化回路の構成を表すマッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段を含み、既存のマッピングデータオブジェクトを前記変更データとして、前記暗号化回路を変更することを特徴とする請求項１記載の暗号化装置。

【請求項３】　前記変更手段は、ハードウェア記述言語により記述されたライブラリをコンパイルして、前記暗号化回路の構成を表すマッピングデータオブジェクトを生成するコンパイラ手段と、該マッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段とを含み、既存のライブラリを前記変更データとして読み込んでコンパイルし、前記暗号化回路を変更することを特徴とする請求項１記載の暗号化装置。

【請求項４】　前記変更手段は、あらかじめ決められた暗号化のアルゴリズムを記述した暗号化アルゴリズムファイルを記憶するデータベース手段と、ハードウェア記述言語により記述されたライブラリをコンパイルして、前記暗号化回路の構成を表すマッピングデータオブジェクトを生成するコンパイラ手段と、該マッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段とを含み、外部から前記変更データとして与えられた設定データに基づいて、対応する暗号化アルゴリズムファイルを検索し、該対応する暗号化アルゴリズムファイルに記述されたライブラリを用いて、前記暗号化回路を変更することを特徴とする請求項１記載の暗号化装置。

【請求項５】　通信ネットワークに接続するネットワーク接続手段をさらに備え、前記変更手段は、前記変更データを該ネットワークから読み込むことを特徴とする請求項１記載の暗号化装置。

【請求項６】　前記ネットワーク接続手段は、暗号化された前記変更データを前記ネットワークから受け取り、前記変更手段は、前記暗号化された変更データに基づいて前記暗号化回路を変更することを特徴とする請求項５記載の暗号化装置。

【請求項７】　前記変更手段は、前記暗号化の仕様を定期的に更新することを特徴とする請求項１記載の暗号化装置。

【請求項８】　前記変更手段は、外部からの要請に基づいて、前記暗号化の仕様を更新することを特徴とする請求項１記載の暗号化装置。

【請求項９】　前記変更手段は、被暗号化データの通信経路、該被暗号化データの機密度、および該被暗号化データに対して要求される処理速度のうち、少なくとも１つに応じて、前記暗号化の仕様を変更することを特徴とする請求項１記載の暗号化装置。

【請求項１０】　少なくとも１つ以上のプログラマブル論理素子を含み、該プログラマブル論理素子を用いて、与えられた復号化の仕様に対応する復号化回路を生成する回路手段と、

前記復号化の仕様を変更するための変更データを読み込み、該変更データに基づいて、前記復号化回路を自動的に変更する変更手段とを備えることを特徴とする復号化装置。

【請求項１１】　前記変更手段は、前記復号化回路の構成を表すマッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段を含み、既存のマッピングデータオブジェクトを前記変更データとして、前記復号化回路を変更することを特徴とする請求項１０記載の復号化装置。

【請求項１２】　前記変更手段は、ハードウェア記述言語により記述されたライブラリをコンパイルして、前記復号化回路の構成を表すマッピングデータオブジェクトを生成するコンパイラ手段と、該マッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段とを含み、既存のライブラリを前記変更データとして読み込んでコンパイルし、前記復号化回路を変更することを特徴とする請求項１０記載の復号化装置。

【請求項１３】　前記変更手段は、あらかじめ決められた復号化のアルゴリズムを記述した復号化アルゴリズムファイルを記憶するデータベース手段と、ハードウェア記述言語により記述されたライブラリをコンパイルして、前記復号化回路の構成を表すマッピングデータオブジェクトを生成するコンパイラ手段と、該マッピングデータオブジェクトを前記プログラマブル論理素子に書き込むコンフィギュレーション手段とを含み、外部から前記変更データとして与えられた設定データに基づいて、対応する復号化アルゴリズムファイルを検索し、該対応する復号化アルゴリズムファイルに記述されたライブラリを用いて、前記復号化回路を変更することを特徴とする請求項１０記載の復号化装置。

【請求項１４】　通信ネットワークに接続するネットワーク接続手段をさらに備え、前記変更手段は、前記変更データを該ネットワークから読み込むことを特徴とする請求項１０記載の復号化装置。

【請求項１５】　前記ネットワーク接続手段は、復号化された前記変更データを前記ネットワークから受け取り、前記変更手段は、前記復号化された変更データに基

づいて前記復号化回路を変更することを特徴とする請求項１４記載の復号化装置。

【請求項１６】　前記変更手段は、前記復号化の仕様を定期的に更新することを特徴とする請求項１０記載の復号化装置。

【請求項１７】　前記変更手段は、外部からの要請に基づいて、前記復号化の仕様を更新することを特徴とする請求項１０記載の復号化装置。

【請求項１８】　前記変更手段は、被復号化データの通信経路、該被復号化データの機密度、および該被復号化データに対して要求される処理速度のうち、少なくとも１つに応じて、前記復号化の仕様を変更することを特徴とする請求項１０記載の復号化装置。

【請求項１９】　少なくとも１つ以上のプログラマブル論理素子を含み、該プログラマブル論理素子を用いて、与えられた仕様に対応する回路を生成する回路手段と、前記回路の仕様を変更するための変更データであって、暗号化と復号化のいずれか一方の仕様を表す該変更データを読み込み、該変更データに基づいて、前記回路を自動的に変更する変更手段とを備えることを特徴とする暗号処理装置。

【請求項２０】　通信ネットワークを介して暗号化されたデータをやり取りする通信システムのための暗号処理システムであって、
少なくとも１つ以上のプログラマブル論理素子を含み、与えられた暗号化の仕様に対応する暗号化回路を生成する暗号化回路手段と、
前記暗号化の仕様を変更するための暗号化変更データを読み込み、該暗号化変更データに基づいて、前記暗号化回路を自動的に変更する暗号化変更手段と、
少なくとも１つ以上のプログラマブル論理素子を含み、与えられた復号化の仕様に対応する復号化回路を生成する復号化回路手段と、
前記復号化の仕様を変更するための復号化変更データを読み込み、該復号化変更データに基づいて、前記復号化回路を自動的に変更する復号化変更手段とを備えることを特徴とする暗号処理システム。

【請求項２１】　少なくとも１つ以上のプログラマブル論理素子を用いて、与えられた暗号化の仕様に対応する暗号化回路を生成し、
前記暗号化の仕様を変更するための変更データを読み込み、該変更データに基づいて、前記暗号化回路を自動的に変更することを特徴とする暗号化方法。

【請求項２２】　少なくとも１つ以上のプログラマブル論理素子を用いて、与えられた復号化の仕様に対応する復号化回路を生成し、
前記復号化の仕様を変更するための変更データを読み込み、該変更データに基づいて、前記復号化回路を自動的に変更することを特徴とする復号化方法。

【請求項２３】　少なくとも１つ以上のプログラマブル論理素子を用いて、与えられた仕様に対応する回路を生成し、
前記回路の仕様を変更するための変更データであって、暗号化と復号化のいずれか一方の仕様を表す該変更データを読み込み、該変更データに基づいて、前記回路を自動的に変更することを特徴とする暗号処理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ファイルやメイルの暗号化、メッセージやユーザの認証などの情報セキュリティー一般に広く用いられる暗号技術に係り、情報を暗号化する暗号化装置、暗号化された情報を復号化する復号化装置、および暗号化／復号化方法に関する。

【０００２】

【従来の技術】現在用いられている暗号には大きく分けて、秘密鍵暗号と公開鍵暗号とがある。ここでは、まずこれらの代表として、現在最も広汎に採用されているＤＥＳ（Data Encryption Standard）暗号とＲＳＡ（Rivest-Shamir-Adleman）暗号をそれぞれ例にとり、暗号化のアルゴリズムを説明する。

【０００３】まず、ＤＥＳは、米国を中心として採用されている代表的な秘密鍵暗号化アルゴリズムの規格である。ＤＥＳの暗号化アルゴリズムでは、数値化された平文（plaintext）データを例えば６４ビットの固定長ブロックに分割し、そのブロック単位で秘密鍵を用いたさまざまな演算を行うことで、平文データの暗号化を行う。この秘密鍵は、被暗号化データである平文データと同じビット長である。

【０００４】図１６は、ブロック長が６４ビットの場合のＤＥＳの暗号化アルゴリズムの概要を示している。図１６において、６４ビットの暗号化鍵は縮約転置１を施されて、１段目の処理に入力される。ここで、縮約転置１とは、入力データの１部を除いて残りの部分を転置する操作を意味し、転置とは、データの部分的な入れ替え操作を意味する。

【０００５】転置された暗号化鍵は前半と後半の２つの部分に分割され、それぞれの部分に巡回シフト２が施される。巡回シフト２とは、入力データを左または右にサイクリックにシフトする操作を意味する。巡回シフト２の後、さらに縮約転置３が施される。

【０００６】また、６４ビットの平文は、転置４が施された後、前半と後半の２つの部分に分割されて１段目の処理に入力される。そして、その片方には、縮約転置３の後の暗号化鍵を用いた非線形変換５が施されて、加算６においてもう一方と加算される。このような処理がm段目まで繰り返され、m段目の処理の結果に転置７が施されて６４ビットの暗号文（cryptogram）となる。

【０００７】ＤＥＳの復号化アルゴリズムも、図１６の暗号化アルゴリズムとほとんど同じであるが、巡回シフト２においては、暗号化アルゴリズムと逆向きにデータをシフトする必要がある。

【０００８】次に、ＲＳＡの暗号化アルゴリズムは、非常に強力な公開鍵暗号化アルゴリズムであり、データの暗号化だけでなく、メッセージやユーザの認証も行うことができる。このアルゴリズムでは、公開鍵と秘密鍵の二つの暗号化鍵を使用する。公開鍵は、文書やネットワーク上のデータの形で公開され、誰でもアクセス可能な状態に置かれるが、秘密鍵は、使用者が厳密に保管する必要がある。

【０００９】ＲＳＡの暗号化アルゴリズムは、数論的な

$$n = p \cdot q \qquad (1)$$

$$e \cdot d \equiv 1 \pmod{(p-1)(q-1)} \qquad (2)$$

ここで、ｐとｑは素数である。（２）式は、法（ｐ－１）（ｑ－１）の下での合同式（congruence）であって、ｅ・ｄと１は（ｐ－１）（ｑ－１）を法として合同であることを表している。言い換えれば、ｅ・ｄ－１は、（ｐ－１）（ｑ－１）で割り切れるということである。また、ｅ＜ｎであり、ｅと（ｐ－１）（ｑ－１）は互いに素である。

【００１１】まず、暗号化８において、平文Ｍは法ｎの下でｅ乗され、暗号文Ｃに変換される。すなわち、

$$C \equiv M^e \pmod{n} \qquad (3)$$

となる。そして、復号化９においては、暗号文Ｃは法ｎの下でｄ乗されて、平文Ｍに戻される。すなわち、

$$M \equiv C^d \pmod{n} \qquad (4)$$

となる。

【００１２】このようにして暗号化された暗号文Ｃを解読するには、秘密鍵ｄの値を知る必要があるが、そのためにはｎを素因数分解して、素数ｐとｑを求めなければならない。しかし、ｎが非常に大きな数の場合、現在の計算機パワーでは、現実的な処理時間内で素因数分解を行うことができない。

【００１３】

【発明が解決しようとする課題】しかしながら、従来の暗号化／復号化技術には次のような問題がある。上述のような堅牢性の高い暗号化アルゴリズムは、ビット長の比較的長い暗号化鍵を用いて複雑な演算を行うため、一般に処理速度が遅く、ソフトウェアでの実装は小規模データの処理などに用途が限定されてしまう。特に、ネットワークで結ばれた情報処理装置間で、通信しながら暗号化／復号化を行うようなリアルタイム処理には、ほとんど実用にならない。

【００１４】そこで、暗号化アルゴリズムをハードウェア的に実現したチップがすでに販売されているが、使用可能なアルゴリズムや暗号化鍵のビット長などが固定されており、実用上のフレキシビリティに欠けている。

【００１５】特に、ＤＥＳのブロック長や、ＲＳＡの暗号化鍵のビット長などは、暗号化アルゴリズムの堅牢性に強く関係している。これらが小さすぎると、巧妙な手法や強力な計算機の援用により、暗号が破られる可能性が高くなる。したがって、セキュリティを確保するためには、機密の程度やその時代の計算機パワーに合わせて、充分な設定値を採用する必要がある。

【００１６】本発明の課題は、必要な機密度などの条件に応じて、フレキシブルにアルゴリズムを変更することや演算を暗号化や復号化に用いており、巨大な素数の素因数分解が非常に困難であることを、暗号の堅牢性の基礎に置いている。

【００１０】図１７は、公開鍵で暗号化したデータを秘密鍵で復号化する場合のＲＳＡの暗号化／復号化アルゴリズムの概要を示している。図１７において、暗号化８で用いられる暗号化鍵（ｅ，ｎ）は、公開された特定の整数ｅとｎの組であり、復号化９で用いられる復号化鍵（ｄ，ｎ）は、同じｎと非公開の整数ｄの組である。これらの数は、次式に基づいて決められる。

【００１７】

【課題を解決するための手段】図１は、本発明の暗号化／復号化装置の原理図である。図１の暗号化／復号化装置は、回路手段１１および変更手段１２を備える。

【００１８】回路手段１１は、少なくとも１つ以上のプログラマブル論理素子１３を含み、それらのプログラマブル論理素子１３を用いて、与えられた暗号化／復号化の仕様に対応する暗号化／復号化回路を生成する。

【００１９】変更手段１２は、上記暗号化／復号化の仕様を変更するための変更データを読み込み、その変更データに基づいて、上記暗号化／復号化回路を自動的に変更する。

【００２０】プログラマブル論理素子１３としては、例えばＦＰＧＡ（field programmablegate array）が用いられ、回路手段１１は、プログラマブル論理素子１３の仕様を、その内部構成を表すマッピングデータオブジェクトなどの形で読み込むことで、暗号化／復号化の仕様に対応する暗号化／復号化回路を生成する。

【００２１】変更手段１２は、マッピングデータオブジェクトなどの形で与えられる上記変更データを読み込み、その変更データをプログラマブル論理素子１３にダイナミックに反映させることで、暗号化／復号化回路内のゲート配置や配線などを変更する。

【００２２】こうして変更された暗号化／復号化回路は、入力される被暗号化データ（平文）／被復号化データ（暗号文）を、変更後の仕様に従って、暗号化データ（暗号文）／復号化データ（平文）に変換する。

【００２３】このような暗号化／復号化装置によれば、

暗号化／復号化回路の内部構成が可変であるため、データの機密度や用途などに応じて、暗号化／復号化回路の仕様をダイナミックに変更することが可能になる。また、その変更は、与えられた変更データをもとにして、自動的に行われる。

【００２４】また、変更手段１２にマッピングデータオブジェクトを自動生成する機能を持たせれば、暗号化／復号化アルゴリズムの種類などを変更データとして指定するだけで、暗号化／復号化回路の仕様を自動的に変更することも可能である。

【００２５】したがって、特に回路設計の知識を持たないユーザであっても、簡単に暗号化／復号化回路を変更することができ、フレキシビリティに富んだ装置が実現される。さらに、暗号化／復号化の動作自身はハードウェアにより実行されるので、ソフトウェアによる処理に比べてはるかに高速である。

【００２６】例えば、図１の回路手段１１は、実施形態の図２におけるプログラマブル論理素子／装置３０およびその周辺回路（不図示）に対応し、変更手段１２は、ホストＣＰＵ（中央処理装置）２１、ハードウェア記述言語ライブラリ生成装置２５、ハードウェア記述言語コンパイラ２７、およびコンフィギュレーション装置２９に対応する。

【００２７】

【発明の実施の形態】以下、図面を参照しながら、本発明の実施の形態を詳細に説明する。本発明の暗号化／復号化装置では、主としてＦＰＧＡのようなプログラマブル論理素子／装置を用いて、暗号化／復号化アルゴリズムを実装する。ここで、論理素子とは１つの半導体チップを意味し、論理装置とは２つ以上の半導体チップを含むような基板や装置を意味する。また、プログラマブル論理素子／装置とは、ユーザ自身が書き込み機（アクティベータ）と設計ソフトウェアを用いて、短時間で試作できるような論理素子／装置である。

【００２８】本発明では、ＦＰＧＡ以外に、ＦＰＧＡの１／１０程度の回路規模を持つＰＬＤ（programmable logic device）、ＰＬＡ（programmable logic array）、ＡＳＩＣ（application specific integrated circuit）など、任意のプログラマブル論理素子／装置を用いることができる。

【００２９】プログラマブル論理素子／装置は、ユーザ自身が内部の論理を作成／変更できるため、実現される暗号化／復号化装置の仕様（スペック）は、既存のマッピングデータ、ネットワーク経由、自動生成などの方法によりダイナミックに変更可能となる。このため、データの機密度や用途に応じて、ユーザが暗号化／復号化装置をカスタマイズすることができる。

【００３０】このようなシステムを採用することにより、複数のアルゴリズムや複数のブロック長や複数の鍵のビット長に対して、ダイナミックに対応可能な暗号化／復号化装置が実現される。また、その装置本体はハードウェア的に実現されているので、暗号化／復号化の実行時においては、大規模データの処理やリアルタイム処理にも充分な効率が確保できる。

【００３１】図２は、このような暗号化／復号化装置の構成を示している。図２の暗号化／復号化装置は、ホストＣＰＵ２１、データベース２３、ハードウェア記述言語ライブラリ生成装置２５、ハードウェア記述言語コンパイラ２７、コンフィギュレーション装置２９、プログラマブル論理素子／装置３０、およびこれらの各装置を結ぶバス２０を備え、動作モードとしてコンフィギュレーションフェーズと実行フェーズを持っている。

【００３２】コンフィギュレーションフェーズにおいて、ユーザから特定の暗号化／復号化回路の作成を指示されると、ハードウェア記述言語コンパイラ２７は、まず対応する暗号化／復号化アルゴリズムをハードウェア記述言語で記述した暗号化／復号化アルゴリズムファイル２４を、データベース２３から取り出す。暗号化／復号化回路作成の指示は、外部装置またはネットワーク２２からも受け付けることができる。

【００３３】ここで、ハードウェア記述言語（ＨＤＬ：hardware desciption language）とは、プログラマブル論理素子／装置３０の内部構成を記述するための言語で、ＶＨＤＬ（ＶＨＳＩＣ－ＨＤＬ：very high-speed integrated circuit hardwaredesciption language）や、それを改良したＶｅｒｉｌｏｇ－ＨＤＬなどがある。例えば、プログラマブル論理素子／装置３０のピン番号やファンクション（ロジック）などが、ハードウェア記述言語により記述される。

【００３４】次に、ハードウェア記述言語コンパイラ２７は、ハードウェア記述言語ライブラリ生成装置２５が生成したハードウェア記述言語ライブラリ２６を用いて、暗号化／復号化アルゴリズムファイル２４をコンパイルし、マッピングデータオブジェクト２８を生成する。

【００３５】マッピングデータオブジェクト２８は、バイナリデータのビット列から成り、プログラマブル論理素子／装置３０の内部のゲート配置や配線などを表す。ＦＰＧＡを用いた場合、そのテクノロジーに適合した形式のバイナリデータが用いられ、それがＦＰＧＡにダウンロードされると、特定の機能が設定される。

【００３６】コンフィギュレーション装置２９は、マッピングデータオブジェクト２８をプログラマブル論理素子／装置３０に書き込んで、配線やロジックを形成し、ユーザの指示に対応する特定の暗号化／復号化回路を作成する。

【００３７】本実施形態においては、ハードウェア記述言語ライブラリ生成装置２５、ハードウェア記述言語コンパイラ２７、およびコンフィギュレーション装置２９の各機能は、ホストＣＰＵ２１が実行するプログラムにより実現される。

【００３８】図３は、図２の暗号化／復号化装置を実現する情報処理装置の構成図である。図３の情報処理装置は、ＣＰＵ３１、メモリ３２、入力装置３３、出力装置３４、外部記憶装置３５、媒体駆動装置３６、ネットワーク接続装置３７を備え、それらの各装置はバス３８により互いに結合されている。

【００３９】ＣＰＵ３１はホストＣＰＵ２１に対応し、メモリ３２に格納されたプログラムを実行して、ハードウェア記述言語ライブラリ生成装置２５、ハードウェア記述言語コンパイラ２７、およびコンフィギュレーション装置２９の各機能を実現する。メモリ３２としては、例えばＲＯＭ（read only memory）、ＲＡＭ（randomaccess memory）などが用いられる。

【００４０】入力装置３３は、例えばキーボード、ポインティングデバイスなどに相当し、ユーザからの指示の入力に用いられる。また、出力装置３４は、表示装置やプリンタなどに相当し、メッセージや処理結果などの出力に用いられる。

【００４１】外部記憶装置３５は、例えば、磁気ディスク装置、光ディスク装置、光磁気ディスク装置などであり、プログラムやデータを保存することができる。また、暗号化／復号化アルゴリズムファイル２４、ハードウェア記述言語ライブラリ２６、マッピングデータオブジェクト２８などを保存するデータベース２３としても使用される。

【００４２】媒体駆動装置３６は、可搬記憶媒体３９を駆動し、その記憶内容にアクセスすることができる。可搬記憶媒体３９としては、メモリカード、フロッピーディスク、ＣＤ－ＲＯＭ（compact disk read only memory）、光ディスク、光磁気ディスクなど、任意の計算機読み出し可能記憶媒体を使用することができる。この可搬記憶媒体３９には、データのほかに、図２の暗号化／復号化装置の処理を行うプログラムが格納される。

【００４３】ネットワーク接続装置３７は、ＬＡＮ（local area network）などの任意の通信ネットワークに接続され、通信に伴うデータ変換等を行う。暗号化／復号化装置は、ネットワーク接続装置３７を介して、外部の情報処理装置などからデータやプログラムを受け取ることができる。

【００４４】例えば、ＤＥＳの暗号化／復号化アルゴリズムを実装する場合、基本ロジックのハードウェア記述言語ライブラリ２６として、１６／３２／６４ビット加算器、１６／３２／６４ビット減算器、８／１６／３２／６４／１２８ビットレジスタ、８／１６／３２／６４／１２８ビット左右シフトレジスタ、１６／３２／６４／１２８ビットインクリメントカウンタ、１６／３２／６４／１２８ビットデクリメントカウンタ、１６／３２／６４ビットＤＥＳ関数発生器、クロック回路、論理和回路、および論理積回路があらかじめ生成され、データベース２３に保存される。

【００４５】一例として、１６ビットインクリメントカウンタをＶｅｒｉｌｏｇ－ＨＤＬで記述すると、図４のようになる。図４のハードウェア記述言語ライブラリ２６では、クロックの立ち上がりでカウント値ｑがインクリメントされることが記されている。

【００４６】また、ＲＳＡの暗号化／復号化アルゴリズムを実装する場合、基本ロジックのハードウェア記述言語ライブラリ２６として、１６／３２／６４／１２８ビット乗算器、１６／３２／６４ビット加算器、１６／３２／６４ビット減算器、８／１６／３２／６４／１２８ビットレジスタ、１６／３２／６４／１２８ビットインクリメントカウンタ、１６／３２／６４／１２８ビットデクリメントカウンタ、クロック回路、論理和回路、および論理積回路があらかじめ生成され、データベース２３に保存される。

【００４７】図２の暗号化／復号化装置にコンフィギュレーションの指示を与える際、暗号化／復号化アルゴリズムの種類、暗号化／復号化鍵のビット長などの設定データを、コマンドの形式で指定する必要がある。

【００４８】例えば、ＲＳＡの暗号化の場合、アルゴリズムの種類としてＲＳＡが指定され、暗号化鍵（ｅ，ｎ）のビット長および暗号化鍵の値が設定される。Ｖｅｒｉｌｏｇ－ＨＤＬによる記述では、配線のビット幅ｗｉｒｅの指定が数値として必要になるので、ホストＣＰＵ２１は、この数値を設定データから生成して、暗号化アルゴリズムファイル２４中のコードに埋め込む。

【００４９】図５は、ビット幅の数値が埋め込まれた暗号化アルゴリズムファイル２４の例を示している。図５のファイルにおいて、行Ｌ１、Ｌ２、Ｌ３の位置に、それぞれ、平文Ｍと暗号文Ｃのビット幅ｂ１＝１５、暗号化鍵ｅのビット幅ｂ２＝７、暗号化鍵ｎのビット幅ｂ３＝６３が記されている。

【００５０】ハードウェア記述言語コンパイラ２７は、このような配線情報に基づいて選択したハードウェア記述言語ライブラリ２６を、暗号化アルゴリズムファイル２４の記述に従って合成して、マッピングデータオブジェクト２８を生成する。

【００５１】次に、図６を参照しながら、コンフィギュレーションフェーズにおける暗号化／復号化回路の作成処理のフローを説明する。図６は、外部から与えられた設定データに基づいて、主としてＲＳＡの暗号化回路を作成する場合のフローチャートであるが、他の暗号化／復号化回路作成の場合も基本的に同様である。

【００５２】図６において処理が開始されると、ハードウェア記述言語コンパイラ２７は、まず指定された暗号化アルゴリズムの種類と、暗号化鍵のビット長と、暗号化鍵の数値とを設定データとして設定し（ステップＳ１、Ｓ２、Ｓ３）、ハードウェア記述言語で記述された対応する暗号化アルゴリズムファイル２４を、データベ

ース２３から自動的に検索する（ステップＳ４）。そして、検索した暗号化アルゴリズムファイル２４の要数コードに、設定データの具体値を入力する（ステップＳ５）。

【００５３】次に、ハードウェア記述言語ライブラリ２６を利用して、暗号化アルゴリズムファイル２４をコンパイルする（ステップＳ６）、これにより、プログラマブル論理素子／装置３０の内部の配置や配線が最適化され、設定データにより指定された特定の暗号化回路のマッピングデータオブジェクト２８が生成される（ステップＳ７）。

【００５４】次に、コンフィギュレーション装置２９は、プログラマブル論理素子／装置３０の周辺回路（不図示）のタイミング信号を発生させ（ステップＳ８）、プログラマブル論理素子／装置３０にマッピングデータオブジェクト２８をダウンロードする。こうして、プログラマブル論理素子／装置３０の配置や配線が作成され（ステップＳ９）、処理が終了する。

【００５５】このように、コンフィギュレーションフェーズにおいては、簡単な設定データを指定するだけで、プログラマブル論理素子／装置３０のプログラミングが自動的に行われるため、設計能力のないユーザでも暗号化／復号化回路を作成することができる。また、設計者にとっても、設計ソフトウェアを用いて回路全体を設計する必要がなく、より短時間で暗号化／復号化回路を作成することができる。

【００５６】図６の処理では、外部から与えられた設定データに基づいて暗号化／復号化回路の仕様を自動生成しているが、他の方法で仕様を変更してもよい。例えば、暗号化鍵のビット長などの具体値が既に埋め込まれた暗号化／復号化アルゴリズムファイル２４を保存しておき、それに基づいてコンパイルを行ってもよい。

【００５７】また、暗号化／復号化アルゴリズムファイル２４を利用せずに、データベース２３に保存されている既存のハードウェア記述言語ライブラリ２６をコンパイルするだけで仕様を変更したり、既存のマッピングデータオブジェクト２８を直接ダウンロードしてそれを変更したりすることもできる。さらに、ネットワーク経由で読み込んだハードウェア記述言語ライブラリ２６やマッピングデータオブジェクト２８を用いて、暗号化／復号化回路の仕様を変更することも可能である。

【００５８】設定データ、暗号化／復号化アルゴリズムファイル２４、ハードウェア記述言語ライブラリ２６、マッピングデータオブジェクト２８のように、暗号化／復号化回路の仕様を変更するために使用され得る情報が、上述の変更データに相当する。

【００５９】一方、実行フェーズにおいては、図７に示すように、コンフィギュレーション済みのプログラマブル論理素子／装置３０は、ホストＣＰＵ２１またはネットワーク４０から平文／暗号文を受け取り、その暗号化／復号化を行う。そして、得られた暗号文／平文は、ホストＣＰＵ２１またはネットワーク４０に出力される。

【００６０】プログラマブル論理素子／装置３０は、ネットワーク４０との間のデータの入出力を、ＴＣＰ／ＩＰ（transmission control protocol/internet protocol）などのプロトコルにより、直接またはホストＣＰＵ２１経由で行うことができる。直接データの入出力を行う場合は、プログラマブル論理素子／装置３０は、ＴＣＰ／ＩＰ制御用のハードウェア（不図示）を介して、ネットワーク４０と接続される。

【００６１】図７のプログラマブル論理素子／装置３０は、設定データの仕様に適合する平文／暗号文を暗号化／復号化するハードウェア回路であり、ソフトウェアを利用した暗号化／復号化処理に比べて、はるかに高速に暗号化／復号化を実行する。このため、ネットワーク４０との間でやり取りされるデータのリアルタイム処理に適している。

【００６２】次に、図８から図１１までを参照しながら、プログラマブル論理素子／装置３０を用いて作成される暗号化／復号化回路の例を説明する。図８および図９は、ＤＥＳの暗号化回路の例を示している。図８は、制御用のタイミング発生と平文の変換を行う回路部分を示しており、図９は、暗号化鍵の変換を行う回路部分を示している。

【００６３】コンフィギュレーションフェーズでは、まず入力／出力のエリアとして、図８に示されるように、入力文字データR_0とL_0を格納するレジスタ４１と４２、および暗号化の繰り返し段数mを指定するレジスタ４９が生成される。また、処理の開始と停止を通知するＳＴＡＲＴ／ＳＴＯＰ信号を格納するレジスタ４７、処理の終了を表すＥＮＤフラグを通知するレジスタ４８、およびＤＥＳの暗号化が完了した文字データが格納されるレジスタ４５、４６も生成される。

【００６４】一方、内部回路としては、ＤＥＳの暗号化アルゴリズムを実現するために、m個の暗号化鍵K_1、K_2、・・・、K_mが設定されるレジスタ４３－１、４３－２、・・・、４３－m、ＤＥＳ関数発生器４４－１、４４－２、・・・、４４－m、クロック回路５０、減算カウンタ５１、およびＯＲ回路５２が定義される。

【００６５】また、図９に示されるように、暗号化鍵に対応する乱数を発生する乱数発生器５３、縮約転置を行うためのビット圧縮回路５４、巡回シフトを行うためのビットシフター５５－１、５５－２、・・・、５５－m、５６－１、５６－２、・・・、５６－m、およびシフト後の暗号化鍵を格納するレジスタ５７－１、５７－２、・・・、５７－mも定義される。

【００６６】実行フェーズでは、繰り返し段数mをレジスタ４９に設定し、入力文字データR_0とL_0をレジスタ４１と４２に設定し、ＳＴＡＲＴ／ＳＴＯＰ信号をＳＴＡＲＴにすると、クロック回路５０によりクロック信号が発生し、減算カウンタ５１、ＤＥＳ関数発生器４４－i、乱数発生器５３、ビット圧縮回路５４、ビットシフター５５－i、５６－i（i＝1，．．．，m）にクロック信号が伝わる。

【００６７】減算カウンタ５１は、繰り返し段数mだけ数えると値０を出力し、ＯＲ回路５２の出力するＨＯＬＤ信号は１から０になる。これにより、クロック回路５０と減算カウンタ５１は停止する。

【００６８】ビット圧縮回路５４は、乱数発生器５３が発生する乱数のビット長を削減し、、ビットシフター５５－i、５６－iは、クロック信号に同期して、圧縮された乱数をシフトし、レジスタ５７－iに入力する。また、ＤＥＳ関数発生器４４－iは、クロック信号に同期して、逐次パイプライン構造により、暗号化鍵K_iを用いた一連の計算を行い、mクロックサイクル後に暗号化が完了した文字データをレジスタ４５、４６に出力する。

【００６９】ＤＥＳの復号化回路は、構成としては図８および図９の暗号化回路と同様であり、暗号化が完了した文字データを入力として、平文の文字データを出力する。図１０は、ＲＳＡの暗号化回路の例を示している。ＲＳＡによる暗号化のコンフィギュレーションフェーズでは、まず入力／出力のエリアとして、公開暗号化鍵eを格納するレジスタ６１と、公開暗号化鍵nを格納するレジスタ６２と、入力としての平文Mを格納するレジスタ６３が生成される。また、処理の開始と停止を通知するＳＴＡＲＴ／ＳＴＯＰ信号を格納するレジスタ６４、処理の終了を表すＥＮＤフラグを通知するレジスタ６５、およびＲＳＡの暗号化が完了した暗号文Ｃが格納されるレジスタ７１も生成される。

【００７０】一方、内部回路としては、ＲＳＡのアルゴリズムを実現するために、クロック回路６６、減算カウンタ６７、ＯＲ回路６８、乗算器６９、および剰余器７０が定義される。

【００７１】実行フェーズでは、暗号化鍵eをレジスタ６１へ、暗号化鍵nをレジスタ６２へセットし、平文Mをレジスタ６３に設定し、ＳＴＡＲＴ／ＳＴＯＰ信号をＳＴＡＲＴにすると、クロック回路６６によりクロック信号が発生し、減算カウンタ６７、乗算器６９、剰余器７０にクロックが伝わる。

【００７２】減算カウンタ６７は、暗号化鍵eの値だけ数えると値０を出力し、ＯＲ回路６８の出力するＨＯＬＤ信号は１から０になる。これにより、クロック回路６６と減算カウンタ６７は停止する。減算カウンタ６７は、最大nまで数えられるようになっている。

【００７３】乗算器６９と剰余器７０は、クロック信号に同期して、（３）式に相当する一連の計算を行う。（３）式の右辺のM^e（ｍｏｄｎ）は、M^eをnで除算したときの剰余と解釈することができるが、これは次式のような展開式により求めることができる。

$$
\begin{aligned}
&M^0 = 1 \\
&M^1 = M \ (\mathrm{mod}\ n) \\
&M^2 = M^1 \cdot M \ (\mathrm{mod}\ n) \\
&\cdots\cdots\cdots\cdots \\
&M^{e-1} = M^{e-2} \cdot M \ (\mathrm{mod}\ n) \\
&M^e = M^{e-1} \cdot M \ (\mathrm{mod}\ n)
\end{aligned}
\qquad (5)
$$

（５）式の右辺はすべて、法nによる除算の剰余を表している。図１０の乗算器６９と剰余器７０は、ＨＯＬＤ信号が０になるまで（５）式に基づく計算を実行し、最終的にM^e（ｍｏｄｎ）を出力する。こうして、暗号文Ｃが生成される。

【００７４】また、ＲＳＡによる復号化のコンフィギュレーションフェーズでは、図１１のような復号化回路が構成される。図１１の回路は、図１０の暗号化回路と同様の構成であるが、暗号化鍵ｅ、ｎと平文Ｍの代わりに、復号化鍵ｄ、ｎと暗号文Ｃが、それぞれレジスタ６１、６２、６３に入力されることになる。また、レジスタ７１からは、暗号文Ｃの代わりに、平文Ｍが出力される。実行フェーズにおける復号化回路の動作は、図１０の暗号化回路と同様である。

【００７５】次に、図１２から図１５までを参照しながら、本発明の暗号化／復号化装置の適用例について説明する。図１２は、暗号化／復号化装置の仕様を定期的に更新する方法を示している。図１２において、プログラマブル論理素子／装置３０は、ネットワーク４０を介して、遠隔地のコンピュータ８１と結ばれている。ノード８２、８３は、ネットワーク４０上に設けられた中継コンピュータや中継器などである。

【００７６】コンピュータ８１はタイマを用いて時間を計測し、一定時間毎に、設定データの変更指示をホストＣＰＵ２１に送信する。これにより、ホストＣＰＵ２１は、設定データを変更して、コンフィギュレーションを再度実行し、暗号化／復号化装置の機能を変更する。このとき、例えばアルゴリズムや鍵が更新される。遠隔地のコンピュータ８１の代わりに、ホストＣＰＵ２１や他のスタンドアロンのＣＰＵで、時間を計測してもよい。

【００７７】このようなシステムにより、一定時間毎に暗号化／復号化装置の仕様を更新することができ、暗号がより解読されにくくなる。図１３は、図１２のシステム構成において、暗号化／復号化装置の仕様を外部からの要請に基づいて更新する方法を示している。図１３において、ホストＣＰＵ２１からの接続要求が遠隔地のコンピュータ８１に伝えられると、コンピュータ８１による接続許可の返信時に、設定データの変更を指示する。

これにより、設定データが変更され、コンフィギュレーションが再度実行されて、暗号化／復号化装置の機能が変化する。

【００７８】このようなシステムにより、暗号化／復号化装置の仕様を、外部から更新することができるようになる。図１４は、暗号化／復号化装置の仕様を被暗号化データの機密度に応じて更新する方法を示している。図１４において、プログラマブル論理素子／装置３０は、ネットワーク４０を介して、遠隔地のコンピュータ８５、８７、８９、９１と結ばれている。ノード８４、８６、８８、９０、９２は、ネットワーク４０上に設けられた中継コンピュータや中継器などである。

【００７９】この暗号化／復号化装置では、安全性をより高めるために、あらかじめアルゴリズムや鍵を複数用意しておき、データの転送経路や要求される機密度によって、使用するアルゴリズムや鍵の種類を変化させる。

【００８０】ここでは、ホストＣＰＵ２１は、コンピュータ８５との通信にＲＳＡアルゴリズムと暗号化鍵ｅ１を使用し、コンピュータ８７との通信にＤＥＳアルゴリズムと暗号化鍵ｋ１を使用し、コンピュータ８９との通信にＲＳＡアルゴリズムと暗号化鍵ｅ２を使用し、コンピュータ９１との通信にＤＥＳアルゴリズムと暗号化鍵ｋ２を使用している。

【００８１】このようなシステムにより、通信経路の安全性やデータの機密度などに応じて、暗号化／復号化装置の仕様を変更することができ、暗号がより解読されにくくなる。

【００８２】図１５は、暗号化／復号化装置の仕様を、必要とされる処理速度に応じて変更する方法を示している。図１５において、プログラマブル論理素子／装置３０、３０′は、ネットワーク４０を介して、遠隔地のコンピュータ９３と結ばれている。ノード９４、９５、９６は、ネットワーク４０上に設けられた中継コンピュータや中継器などである。プログラマブル論理素子／装置３０にはホストＣＰＵ２１が接続され、プログラマブル論理素子／装置３０′にはホストＣＰＵ２１′が接続されている。

【００８３】被暗号化データが、例えば画像のように大量に存在するデータである場合、データ量と要求される処理速度とに応じて、鍵のビット長やＤＥＳアルゴリズムの繰り返し段数などを変化させる。これにより、プログラマブル論理素子／装置３０による小規模な画像データの処理と、プログラマブル論理素子／装置３０′による大規模な画像データの処理とを、一定時間内で終了させることができる。

【００８４】また、図１２から図１５までのシステムにおいて、暗号化／復号化装置の仕様を変更するために必要な変更データを暗号化して、遠隔地のコンピュータ８１、８５、８７、８９、９１からホストＣＰＵ２１、２１′に送信することもできる。

【００８５】この場合、遠隔地のコンピュータ８１、８５、８７、８９、９１には、例えば本発明の暗号化／復号化装置が接続され、それにより変更データが暗号化される。また、受信された暗号化変更データはプログラマブル論理素子／装置３０、３０′により復号化され、その内容に基づいて仕様が変更される。このように、変更データを暗号化してやり取りすることで、仕様変更の事実や更新後の仕様を他人に知られることが防止される。

【００８６】以上説明した実施形態においては、主としてＤＥＳアルゴリズムとＲＳＡアルゴリズムを用いた例を説明したが、これらは一例に過ぎず、本発明の暗号化／復号化装置は、他の任意の暗号化／復号化アルゴリズムを実装することができる。また、設定データの内容はアルゴリズムに応じて変化し、例えば、使用するハードウェア記述言語ライブラリ２６のファイル名を、設定データに直接記述することも可能である。

【００８７】

【発明の効果】本発明によれば、高速かつフレキシブルな暗号化／復号化装置が実現される。これにより、大規模なデータの暗号化／復号化装置やリアルタイムの暗号化／復号化装置を、機密の度合いや用途に応じてエンドユーザがカスタマイズしたり、自動生成したりすることが可能になる。

【図面の簡単な説明】

【図１】本発明の暗号化／復号化装置の原理図である。

【図２】実施形態における暗号化／復号化装置の構成図である。

【図３】情報処理装置の構成図である。

【図４】ライブラリの例を示す図である。

【図５】暗号化アルゴリズムファイルの例を示す図である。

【図６】コンフィギュレーションフェーズにおける処理のフローチャートである。

【図７】実行フェーズを示す図である。

【図８】ＤＥＳの暗号化回路を示す図（その１）である。

【図９】ＤＥＳの暗号化回路を示す図（その２）である。

【図１０】ＲＳＡの暗号化回路を示す図である。

【図１１】ＲＳＡの復号化回路を示す図である。

【図１２】仕様の定期的な更新方法を示す図である。

【図１３】要請に基づく仕様の更新方法を示す図である。

【図１４】機密度に応じた仕様の変更方法を示す図である。

【図１５】処理速度に応じた仕様の変更方法を示す図である。

【図１６】ＤＥＳのアルゴリズムを示す図である。

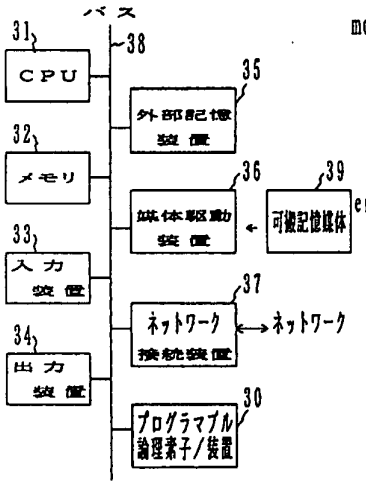
【図１７】ＲＳＡのアルゴリズムを示す図である。

【符号の説明】

１、３　縮約転置
２　巡回シフト
４、７　転置
５　非線形変換
６　加算
８　暗号化
９　復号化
１１　回路手段
１２　変更手段
１３　プログラマブル論理素子
２０、３８　バス
２１、２１′　ホストＣＰＵ
２２　外部装置またはネットワーク
２３　データベース
２４　暗号化／復号化アルゴリズムファイル
２５　ハードウェア記述言語ライブラリ生成装置
２６　ハードウェア記述言語ライブラリ
２７　ハードウェア記述言語コンパイラ
２８　マッピングデータオブジェクト
２９　コンフィギュレーション装置
３０、３０′　プログラマブル論理素子／装置
３１　ＣＰＵ
３２　メモリ
３３　入力装置
３４　出力装置
３５　外部記憶装置
３６　媒体駆動装置
３７　ネットワーク接続装置
３９　可搬記憶媒体
４０　ネットワーク
４１、４２、４３－１、４３－２、４３－３、４３－ｍ、４５、４６、４７、４８、４９、５７－１、５７－２、５７－３、５７－ｍ、６１、６２、６３、６４、６５、７１　レジスタ
４４－１、４４－２、４４－３、４４－ｍ　ＤＥＳ関数発生器
５０、６６　クロック回路
５１、６７　減算カウンタ
５２、６８　ＯＲ回路
５３　乱数発生器
５４　ビット圧縮回路
５５－１、５５－２、５５－３、５５－４、５５－ｍ、５６－１、５６－２、５６－３、５６－４、５６－ｍ　ビットシフター
６９　乗算器
７０　剰余器
８１、８５、８７、８９、９１　コンピュータ
８２、８３、８４、８６、８８、９０、９２、９４、９５、９６　ノード

【図３】

情報処理装置の構成図



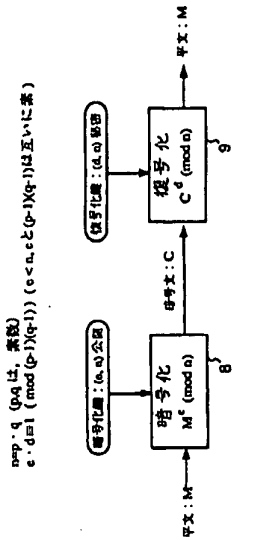
【図４】

ライブラリの例を示す図

```
module Bcount16(q, clk)
  output[15:0] q;
  input clk;
  reg[15:0] q;

  always@(posedge clk)
      q=q+ d1;
endmodule
```

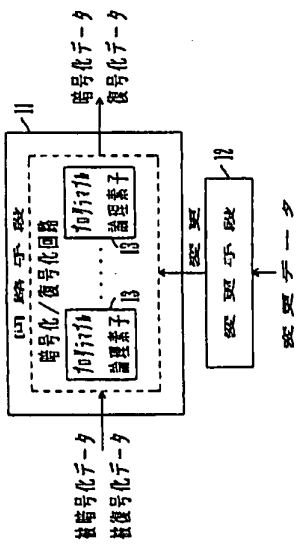
【図１７】

RSAのアルゴリズムを示す図



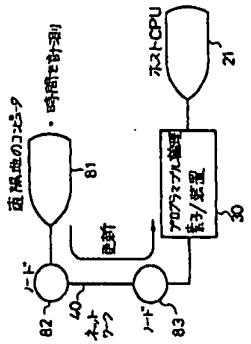
【図1】

本発明の原理図



【図12】

仕様の定期的な更新方法を示す図



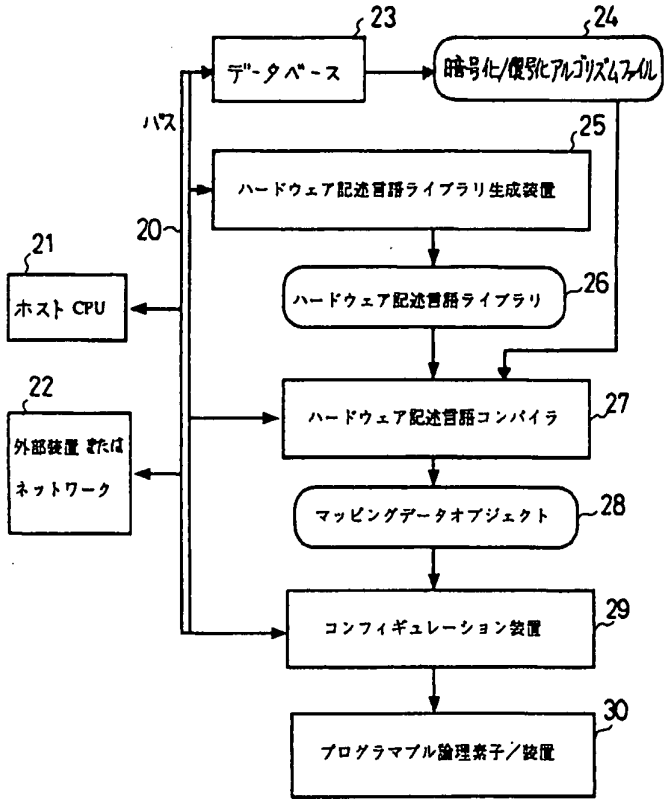
【図5】

暗号化アルゴリズムファイルの例を示す図

```
    module top;
      reg clock, reset, start,  end;
1)    wire [b1:0] M, C;   <-15
2)    wire [b2:0] e;      <- 7
3)    wire [b3:0] n;      <-63

    rsaEnc     enc1(M, C, e, n, clock, reset, start, end);

    endmodule

    module rsaEnc(M, C, e, n, clk, res, st, ed);
       input [b1:0] M;   <-15
       input [b2:0] e;   <- 7
       input [b3:0] n;   <-63
       input  clk, res, st;
       output [b1:0] C; <-15
       output ed;
     {
        integer i;
        always@(posedge clk)
        if (res == 1'b1)
           C = 16'd0;
        else if (st == 1'b1)
                C = 1'b1;
                for (i=0;i<e;i++) {
                    C=(M*C)%n;
                }
                ed=1'b1;

     }
    endmodule
```

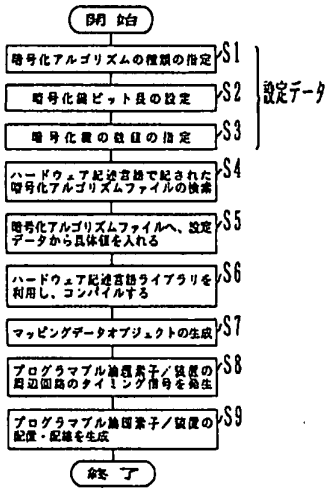

【図2】

暗号化/復号化装置の構成図



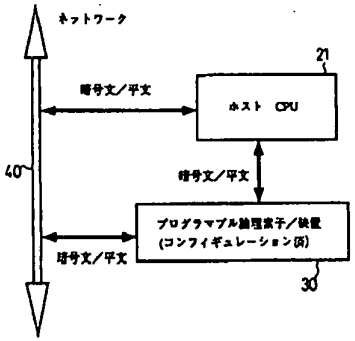
【図6】

コンフィギュレーションフェーズにおける処理のフローチャート



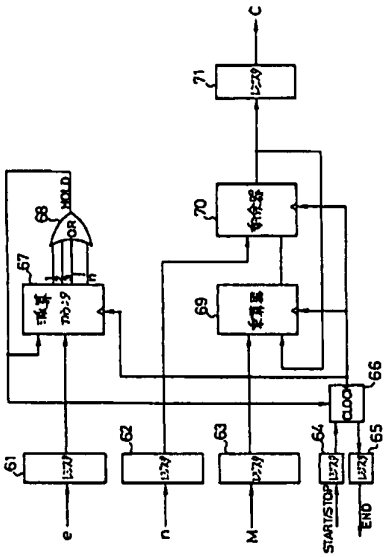
【図7】

実行フェーズを示す図



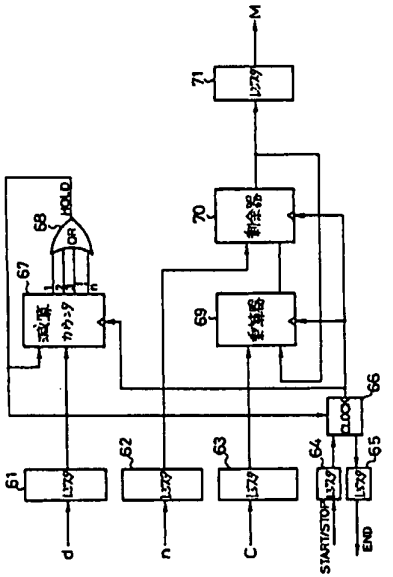
【図10】

RSAの暗号化回路を示す図



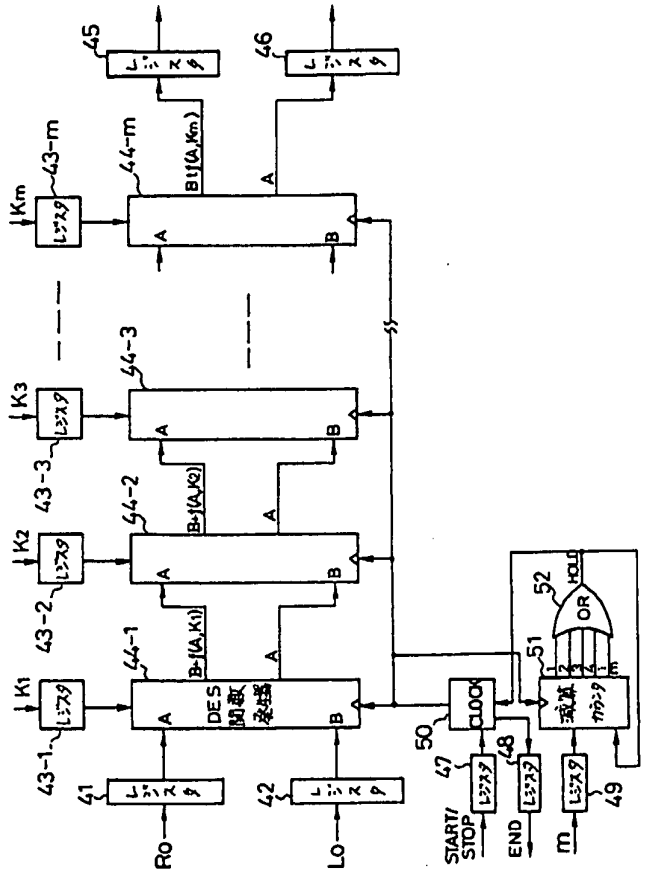
【図11】

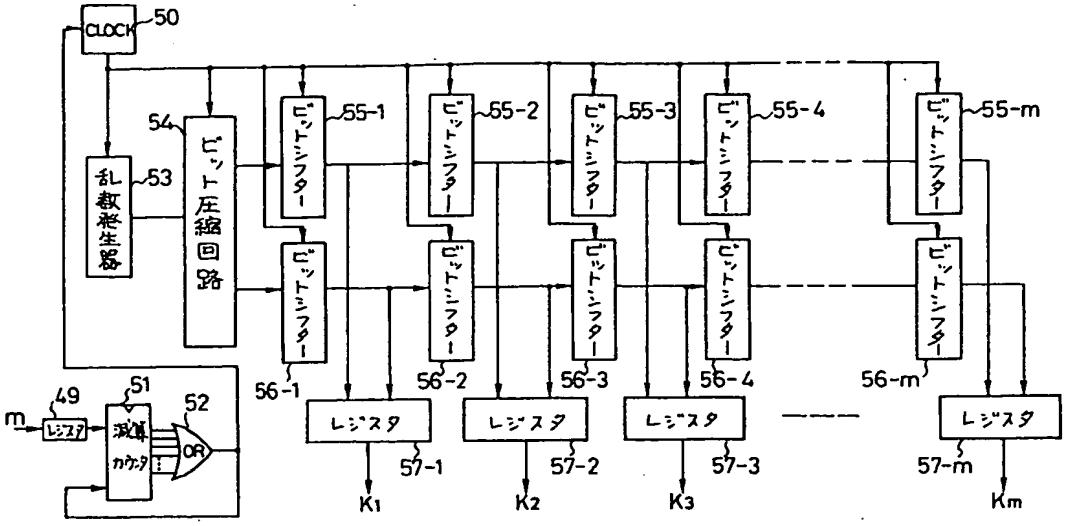
RSAの復号化回路を示す図



【図8】

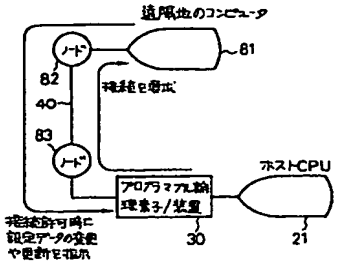
DESの暗号化回路を示す図（その1）





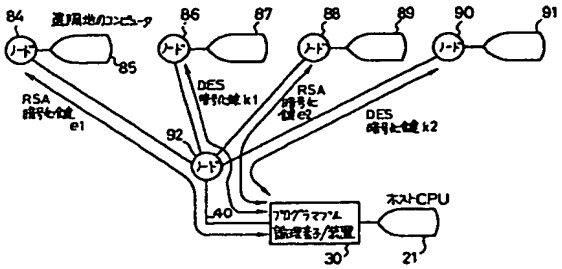
【図１】

DESの暗号化回路を示す図（その２）



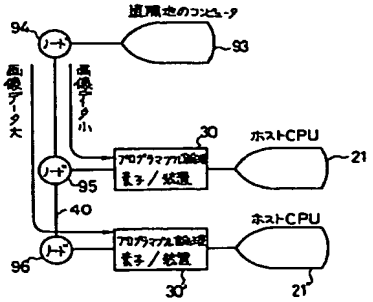
【図１３】

暗号に基づく仕様の更新方法を示す図



【図１４】

機能変更における仕様の変更方法を示す図



【図１５】

処理速度における仕様の変更方法を示す図

【図16】

DESのアルゴリズムを示す図